NPS52-82-007

NAVAL POSTGRADUATE SCHOOL

Monterey, California

DISTRIBUTED VS. CENTRALIZED DATABASE SYSTEMS - TRANSACTION EXECUTION COST AND PERFORMANCE ANALYSIS

Dushan Z. Badal

July 1982

NAVAL POSTGRADUATE SCHOOL
Monterey, California

Rear Admiral J. J. Ekelund
Superintendent

D. A. Schrady
Acting Provost

The work reported herein was supported in part by the Foundation Research Program of the Naval Postgraduate School with funds provided by the Chief of Naval Research.

This report was prepared by:

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE *(When Data Entered)*

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER NPS52-82-007	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE *(and Subtitle)* Distributed vs. Centralized Database Systems - transaction execution cost and performance analysis		5. TYPE OF REPORT & PERIOD COVERED Technical Report 6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Dushan Z. Badal		8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS Naval Postgraduate School Monterey, CA 93940		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS Naval Postgraduate School Monterey, CA 93940		12. REPORT DATE July 1982 13. NUMBER OF PAGES
14. MONITORING AGENCY NAME & ADDRESS*(if different from Controlling Office)*		15. SECURITY CLASS. *(of this report)* UNCLASSIFIED 15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT *(of this Report)*

Approved for public release; distribution unlimited

17. DISTRIBUTION STATEMENT *(of the abstract entered in Block 20, if different from Report)*

18. SUPPLEMENTARY NOTES

A part of this report has been published in The Proceedings of the Second International Symposium of Distributed Databases, September 1 - 3, 1982, Berlin.

19. KEY WORDS *(Continue on reverse side if necessary and identify by block number)*

Distributed databases, performance analysis, cost analysis, long-haul networks, local area networks, distributed computing.

20. ABSTRACT *(Continue on reverse side if necessary and identify by block number)*

The purpose of this paper is twofold. First, we investigate the impact of concurrency control on transaction execution cost and system throughput in centralized and distributed data base systems (DBS) based on slow and fast (local) networks. Second, we show that in terms of transaction execution cost and DBS throughput there are some applications for which any distributed DBS can be more effective than any centralized DBS and vice versa. We also argue that for other applications the decision in favour of distributed or centralized DBS should be based on the comparison of specific DBS systems.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

S/N 0102-LF-014-6601

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE *(When Data Entered)*

Distributed vs. Centralized Database Systems - transaction execution cost and performance analysis

Dushan Z. Badal

Computer Science Department

Naval Postgraduate School

Monterey, CA 93940

Abstract

The purpose of this paper is twofold. First, we investigate the impact of concurrency control on transaction execution cost and system throughput in centralized and distributed data base systems (DBS) based on slow and fast (local) networks. Second, we show that in terms of transaction execution cost and DBS throughput there are some applications for which any distributed DBS can be more effective than any centralized DBS and vice versa. We also argue that for other applications the decision in favour of distributed or centralized DBS should be based on the comparison of specific DBS systems.

1. Introduction.

Distributed database systems (D-DBS) are alleged to provide numerous advantages over centralized database systems (C-DBS). The usual arguments in favour of the D-DBS are:

a. improved user attitude - the distribution of data and processing gives users greater control and autonomy over the data processing

b. improved reliability and availability - because of the partitioning of data and replication of processors and data

c. improved extensibility and modularity

d. decreasing cost of hardware should make D-DBS cost effective

e. D-DBS could provide better performance and perhaps lower transaction execution cost because they have inherent concurrent execution capabilities not available in C-DBS

We expect that in many applications the last consideration, i.e. transaction execution cost and performance, will be the principal factor when deciding between C-DBS and D-DBS. Therefore in this paper, we analyze and compare the transaction execution cost and the performance of C-DBS and D-DBS. We are here interested in two goals. First, we want to investigate the importance or the impact of concurrency control on transaction execution cost and system throughput in C-DBS and D-DBS based on slow and fast (sometimes called local) networks. Second, we are interested in a simple and robust analysis which explains certain intuitive notions about the preferability or suitability of D-DBS or C-DBS for some applications. Our analysis is general , i.e. it is not meant to represent any particular concurrency control mechanism, C-DBS

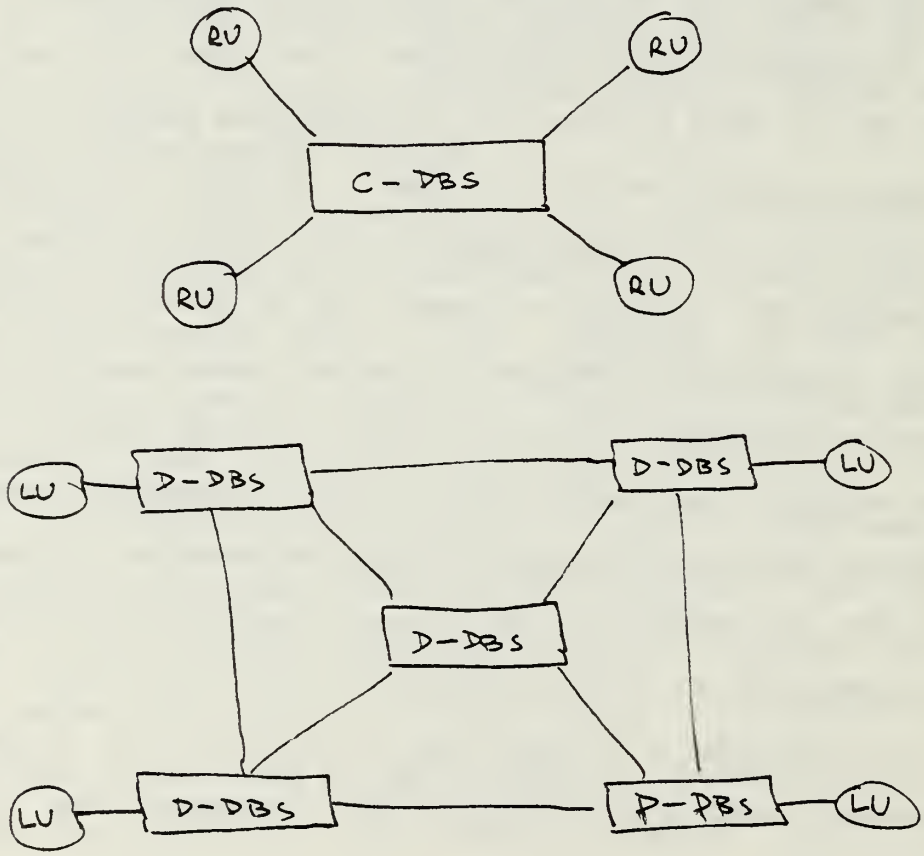
or D-DBS. The analysis however can become specific by substituting proper values, representing specific concurrency control mechanisms or DBS systems, into the derived formulas.

2. Previous work.

It appears that there are no published papers dealing with comparative analysis of distributed and centralized DBS systems. However, recently some work has been done on the performance analysis of distributed DBS (BAD 80, MOL 79, RIE 79) and there are two publications (BUC 79, STE 79) somewhat related to this paper. The paper by Bucci and Streeter (BUC 79) deals with the cost and performance analysis of a single processor with multiple remote terminals. The paper also contains a short sketch of a very simple comparative cost analysis of distributed and centralized DBS. The paper by Stewart (STE 79) describes a discrete simulation modelling tool developed for the performance analysis of IBM SNA system configurations. It is obvious from the paper that this performance analysis tool can be used for already implemented systems as the simulator input requires detailed system implementation information e.g., number of instructions per program, number of I/O, a priority interrupt mechanism, a priority dispatcher, CPU definition, etc. However, it is indicated in the paper that the simulator could be used for the performance analysis of distributed DBS - presumably once it has been implemented.

3. D-DBS vs. C-DBS: Transaction Execution Cost Analysis.

In our analysis, we consider C-DBS with remote users and a D-DBS with similar capabilities in which data and processing power are distributed to the remote users. Figure 1 shows both DBS configurations we analyze in this paper. (RU is remote user and LU is local user).



In this paper the cost is defined in terms of the number of executed instructions, the amount of generated I/O and the number of messages, or any subset of these. We model the average cost of one transaction processing in the C-DBS as consisting of three parts as follows:

$$C_c = C_{csys} + C_{com} + C_{syn} \quad (1)$$

where

C_{csys} is the average cost of executing one transaction in the C-DBS without a concurrency control (CC)

C_{com} is the average (communication) cost of submitting the transaction from the remote user to the C-DBS

C_{syn} is the average CC cost, i.e. the average cost due to synchronization of the transaction

Since any transaction in any DBS is either conflicting (i.e., trying to acquire resources already acquired by some other transaction) or nonconflicting then C_{syn} consists of two types of costs - the cost (C_{cc}) associated with nonconflicting transaction and the cost (C_{confl}) when the transactions interfere, i.e., conflict. We can express C_{syn} as follows:

$$C_{syn} = DI * C_{confl} + C_{cc} \quad (2)$$

where

DI, the degree of interference, is the ratio of the number of conflicting transactions to the number of all transactions

C_{confl} is the average CC conflict resolution cost per conflicting transaction

$DI*C_{confl}$ is the average conflict resolution cost per transaction

C_{cc} is the average CC no-conflict cost per transaction

Transactions in D-DBS can be divided not only into conflicting and nonconflicting (as in C-DBS) but they can be also divided into local and nonlocal or global transactions depending whether they execute at one site only (local transaction) or more than one site (global transactions). Thus the cost of transaction processing in the distributed DBS (D-DBS) can be modeled as the cost of local and nonlocal (or global) transaction executions weighted by the terms reflecting the number of local and global transactions in the system. Then the cost C_d of the transaction processing in D-DBS is:

$$C_d = X*(C_{lsys} + C_{lsyn}) + (1-X)*(C_{gsyn} + C_{gsys}) \quad (3)$$

where

C_{lsys} is the average cost of local transaction execution without considering concurrency control

C_{lsyn} is the average CC cost per local transaction, i.e., one

which needs to access data only at one site

C_{gsys} is the average cost of global transaction execution without considering concurrency control

C_{gsyn} is the average CC cost per global transaction, i.e. one which needs to access data at more than one site of the D-DBS

X is the ratio of the number of local transactions to all transactions

1-X is the ratio of the number of global transactions to all transactions

Let

$$C_{gsys} = C_{lsys} + C_{data} \tag{3a}$$

where

C_{data} is the average (communication) cost of data transfers during the global transaction execution

C_{lsyn} can be further decomposed in a similar manner to C_{syn} in the case of C-DBS:

$$C_{lsyn} = DI_l * C_{lconfl} + C_{lcc} \tag{4}$$

where

DI_l is the degree of interference of local transactions at each site of the D-DBS, i.e. ratio of the number of conflicting local transactions (i.e., local transactions conflicting with local transactions

but not with global transactions) to the total number of local transactions

C_{lcc} is the average CC no-conflict cost per local transaction

C_{lconfl} is the average CC conflict resolution cost per local conflicting transaction.

C_{gsyn} can also be decomposed in similar manner as follows:

$$C_{gsyn} = C_{gcc} + DI_g * C_{gconfl} \quad (5)$$

where

C_{gcc} is the average CC no-conflict cost per global transaction

C_{gconfl} is the average CC conflict resolution cost per global transaction

DI_g is ratio of the number of conflicting transactions (i.e., global transactions conflicting with global and local transactions) to the number of global transactions

Substituting into (1) and (3) from (2), (3a), (4), and (5) we obtain

$$C_c = C_{csys} + C_{com} + DI * C_{confl} + C_{cc} \quad (6)$$

$$C_d = C_{lsys} + X*(DI_l*C_{lconfl} + C_{lcc}) + (1-X)*(DI_g*C_{gconfl} + C_{gcc} + C_{data}) \quad (7)$$

In order to simplify (6) and (7) we assume that the cost of synchronizing local transactions in D-DBS is proportional to the cost of synchronizing the transactions in C-DBS, i.e., we assume that:

$$C_{lconfl} = K_1 * C_{confl}$$

$$C_{lcc} = K_1 * C_{cc} \qquad (8)$$

where $K_1 \neq 1$ if the following applies:

a) processors in D-DBS can not support the same CC mechanism at the same cost as C-DBS, e.g., number of I/O is different, etc.

b) D-DBS uses CC mechanism different from the one used in C-DBS

c) when both of the above apply

We can also assume that the cost of local transaction execution without synchronization in D-DBS is proportional to the cost of transaction execution without synchronization in C-DBS, i.e., we assume:

$$C_{lsys} = K_2 * C_{csys} \qquad (9)$$

where $K_2 \neq 1$ if processors in D-DBS execute local transactions (without CC) at a different cost compared to C-DBS.

We also assume, somewhat arbitrarily, that the degree of global and local transaction interference in D-DBS is proportional to the degree of transaction interference in C-DBS as follows:

$$DI_g = (1-K_3)*DI \qquad (10)$$

$$DI_l = K_3*DI$$

where large K_3 reflects either good D-DBS design and/or applications with strong locality. We feel that the above assumption, whether right or wrong, becomes irrelevant for applications featuring very low degrees of transaction interference. We note here that most of present applications seem to be in that class.

Finally we assume that the cost of synchronizing global transactions in D-DBS is proportional to the cost of synchronizing the transactions in C-DBS, i.e., we assume that:

$$C_{gcc} = K_o * C_{cc} \tag{11}$$

$$C_{gconfl} = K_o * C_{confl}$$

We will refer to a D-DBS which uses slow network (slow compared to secondary memory channels) as a slow D-DBS for which $K_o >> 1$. We will refer to a D-DBS which uses fast network (comparable to secondary memory channels) as a fast (local) D-DBS for which $K_o \approx 1$. Substituting (8), (9), (10) and (11) into (6) and (7) we get

$$C_c = C_{csys} + C_{com} + DI * C_{confl} + C_{cc} \tag{12}$$

$$C_d = K_2 * C_{csys} + X * (K_3 * DI * K_1 * C_{confl} + K_1 * C_{cc}) +$$

$$(1-X) * ((1-K_3) * DI * K_o * C_{confl} + K_o * C_{cc} + C_{data}) \tag{13}$$

We would like to know when the cost of transaction execution is larger in C-DBS compared to the cost of transaction execution in D-DBS. Let

$$C_c \geq C_d \tag{14}$$

Substituting from (12) and (13) into (14) we obtain

$$(1-K_2)*C_{csys} + C_{com} + (1-K_1*K_3*X)*DI*C_{confl} + (1-K_1*X)*C_{cc} >$$

$$(1-X)*(2*K_o*(1-K_3)*DI*C_{confl} + K_o*C_{cc} + C_{data}) \tag{15}$$

If $X \approx 1$, i.e. when almost all transaction in D-DBS are local then (15) reduces to

$$(1-K_2)*C_{csys} + C_{com} + (1-K_1*K_3)*DI*C_{confl} + (1-K_1)*C_{cc} \geq 0 \tag{16}$$

When almost all transactions are local (or equivalently when D-DBS is well designed) we introduce only negligible error by assuming that K3 = 1, i.e., assuming that the degree of interference in C-DBS is the same as in D-DBS. Then (16) can be rewritten as

$$C_{com} \geq (K_2-1)*C_{csys} + (K_1-1)*C_{syn} \tag{17}$$

Let's assume that K1 = 1, i.e. C-DBS and D-DBS execute under the same CC mechanism and the CC cost is the same in both. Then (17) reduces to

$$C_{com} \geq (K_2-1)*C_{csys} \tag{18}$$

If we also assume that K2 = 1, i.e. the cost of transaction execution without CC in D-DBS and C-DBS is the same then

$$C_{com} > 0 \qquad (19)$$

which is always true.

The above result says that in applications where (a) almost all transactions are local, (b) the same CC mechanism is used in C-DBS and D-DBS, (c) local processors in D-DBS do not impose any processing cost penalty compared to C-DBS processor, then the D-DBS regardless of the speed of its network and regardless of the degree of interference will result in lower transaction execution cost. This result which has been derived from the analysis of our model is in a complete accord with our intuition as it is to be expected if the model is realistic.

We come to the same conclusion when $K_2 = 1$ and $K_1 < 1$, (i.e. local processors in D-DBS do not impose any transaction processing cost penalty compared to C-DBS processor and D-DBS executes under different CC mechanism which has lower overhead cost).

Let's assume that $K_1 = K_2$, i.e. the local processors in C-DBS impose the same cost penalty on transactions and CC programs processing. Then from (17) we get

$$C_{com} > (K_1-1)*(C_{syn} + C_{csys}) \qquad (20)$$

From (20) the only scenario on which we can make a general observation on is when $K_1 >> 1$, i.e., when local processors in D-DBS impose heavy processing cost penalty compared to C-DBS processor. Then from (20) we obtain

$$C_{com} >> (C_{syn} + C_{csys}) \quad (21)$$

The conclusion from (21) is that even if $K_1 >> 1$ D-DBS can still result in lower transaction processing cost in applications where users ship large amounts of data per transaction between remote terminals and C-DBS processors, and terminal communication lines are slow and costly and transactions are not computationally intensive.

We note here that our analysis could be made specific by substituting either measured or assumed values for the parameters in our formulas. We avoid doing so as no commercial D-DBS is operational today. Therefore, we rather attempt to use a few reasonable assumptions so that we can simplify our formulas and then make general observations.

Let's assume that DI $\approx$ 0, i.e. very few transactions interfere (that seems to be valid assumption for most applications). Also assume that K1 = K2 = 1,i.e. local processors in D-DBS do not impose any cost penalty on transactions and CC programs processing compared to C-DBS processor. Then from (15) we obtain

$$C_{com} > (1-X)*(C_{data} + C_{cc}*(K_o-1)) \quad (22)$$

For fast D-DBS K_o 1 and (22) reduces to

$$C_{com} > (1-X)*C_{data} \quad (23)$$

For slow D-DBS Ko >> 1 and (22) reduces to

$$C_{com} > (1-X)*(C_{data} + K_o*C_{cc}) \quad (24)$$

The observations we offer on (23) and (24) is that when comparing transaction execution cost in C-DBS and fast D-DBS then the CC mechanism problem is not very important. However, it becomes very important when comparing cost of transaction execution in C-DBS and slow D-DBS.

4. C-DBS vs D-DBS: Performance Analysis.

The second issue we want to investigate in this paper is the impact of concurrency control on throughput of C-DBS and D-DBS based on slow and fast networks. We are also interested in identifying applications for which we can say that any D-DBS will likely outperform any C-DBS and vice versa.

The C-DBS throughput can be derived by considering the fact that system transaction processing rate is decreased by synchronization of transactions. Thus we can express C-DBS throughput Q_C as follows:

$$Q_C = [S_C - (S_{Ccc} + S_{Cconfl})] \tag{25}$$

where

S_C is the basic transaction processing rate of C-DBS which does not have any concurrency control

S_{Ccc} is the fraction of basic transaction processing rate S_C used for synchronization of transactions

S_{Cconfl} is the fraction of S_C used for resolution of conflicts.

We model D-DBS as a set of n processors whose physical dependency due to an underlying network and whose logical and functional dependency due to global transactions are reflected only as a decrease of the throughput of every site in D-DBS. Thus, the throughput of D-DBS can be derived in terms of each node processing rate and its decrease due to the synchronization of local and global transactions as follows:

$$Q_d = n*\{S_L-(S_{Lcc} + DI_L*S_{Lconfl})] + (1-Y)*[C_0*S_L - (S_{Gcc} + DI_G*S_{Gconfl})]\} \quad (26)$$

where

Y is the ratio of local transactions to all transactions

n is the number of sites or nodes in the D-DBS

S_L is the transaction processing rate of each of n nodes without concurrency control

S_{Lcc} is the fraction of S_L used for the synchronization of transactions local to one site

DI_L is the degree of local transaction interference, i.e., the ratio of local interferring transactions to all local transactions

S_{Lconfl} is the fraction of S_L used for the resolution of local transaction conflicts

S_{Gcc} is the fraction of S_L used for the processing, of synchronization messages of global transactions

S_{Gconfl} is the fraction of S_L used for the resolution of global transaction conflicts

DI_G is the degree of global transaction interference, i.e., the ratio of interferring global transactions to all global transactions

C_0 is the ratio of D-DBS average network delay to D-DBS local processor I/O time

We further assume that

$$S_{Gcc} = C_{01}*S_{Lcc}$$
$$S_{Gconfl} = C_{02}*S_{Lconfl}$$

where

$$C_{01} = C_{00}*M_{cc}$$
$$C_{02} = C_{00}*M_{confl}$$

where

C_{00} is the ratio of average network delay to CC message set-up time

M_{cc} is CC no-conflict overhead, i.e., the average number of messages a given CC mechanism requires for synchronization of nonconflicting transactions

M_{confl} is CC conflict overhead, i.e., the average number of messages a given CC mechanism requires for the resolution of transaction conflicts. (An example of CC conflict and no-conflict overhead analysis of several CC mechanisms can be found in (BAD 81)).

We also assume that

$$S_L = C_1*S_C$$
$$S_{Lcc} = C_2*S_{Ccc}$$

$$S_{Lconfl} = C_2 * S_{Cconfl}$$

$$DI_L = C_3 * DI$$

$$DI_G = (1-C_3) * DI$$

We are interested when

$$Q_c \gtrless Q_d \tag{27}$$

Substituting into (27) from (25) and (26) and using the above assumptions we obtain

$$S_C - (S_{Ccc} + DI*C_{confl}) \gtrless n*\{Y*[C_1*S_C - (C_2*S_{Ccc} + C_3*C_2*DI*S_{Cconfl})] + (1-Y)*[C_0*C_1*S_C - (C_{01}*C_2*S_{Ccc} + (1-C_3)*C_{02}*C_2*DI*S_{Cconfl})]\} \tag{28}$$

In order to simplify (28) we will assume that $DI \approx 0$, $C_1 = C_2$ and $C_0 = 1$. Thus we consider applications which have very few conflicting transactions. We also assume that C-DBS and D-DBS either use the same CC mechanisms or if they are different then the decrease of local processor throughput is the same as if they both used the same CC mechanisms ($C_1 = C_2$). Finally we assume fast D-DBS where D-DBS local processor I/O speed is the same as D-DBS network speed ($C_0 = 1$). Substituting these assumptions into (28) leads to

$$\frac{S_C}{S_{Ccc}} * (1 - n*C_1) > 1 - n*C_1*(C_{01} + Y*(1 - C_{01})) \tag{29}$$

We consider three cases when $1 > n*C_1$, $1 = n*C_1$ and $1 < n*C_1$. When $1 > n*C_1$, i.e., $S_C > n*S_L$ then (29) can be rewritten as

$$\frac{S_C}{S_{Ccc}} \geq \frac{1 - n*C_1*(Y + C_{01}*(1 - Y))}{1 - n*C_1} \tag{30}$$

However, for any D-DBS

$$\frac{S_C}{S_{Ccc}} \geq 1 \tag{31}$$

and therefore

$$\frac{(1 - n*C_1*(Y + C_{01}*(1 - Y))}{1 - n*C_1} \geq 1 \tag{32}$$

As we assumed $1 - n*C_1 > 0$ (32) reduces to

$$C_{01} \leq 1 \tag{33}$$

Thus if $S_C > n*S_L$, then $Q_c > Q_d$ only if $C_{01} \leq 1$.

When $n*C_1 = 1$ then (29) considering (31) reduces to

$$C_{01} \geq 1 \tag{34}$$

Thus if $S_C = n*S_L$ then $Q_c \geq Q_d$ only if $C_{01} \geq 1$, else $Q_c < Q_d$.

When $1 < n*C_1$, i.e., when $S_C < n*S_L$, then (29) considering (31) reduces to

$$C_{01} \geq 1 \tag{35}$$

Thus if $S_C < n*S_L$, then $Q_c \geq Q_d$ only if $C_{01} \geq 1$, else $Q_d > Q_c$.

From (33), (34) and (35) we can conclude that when comparing

throughput of C-DBS and D-DBS based on fast network then the CC mechanism (i.e. its CC overhead) and its efficient implementation (i.e. efficient message processing) are quite important. This observation is not entirely intuitive in the light of our assumption of very few conflicting transactions, i.e., DI$\approx$0. Of course if there are many conflicting transactions one would expect CC mechanism to be important for D-DBS throughput.

An interesting observation can be made on (35). Even if D-DBS transaction processing rate without CC mechanism is larger than transaction processing rate without CC in C-DBS, a D-DBS with CC mechanism can perform worse than C-DBS with CC mechanism if $C_{01} < 1$, i.e., if either CC mechanism has high no-conflict overhead or it has slow CC message processing.

Let's consider applications where Y$\approx$1, i.e., there are very few global transactions or equivalently almost all transactions are local. In such case we can also assume that $C_3 = 1$. Substituting these assumptions into (27) we obtain

$$S_C*(1 - n*C_1) \geq (S_{Ccc} + DI*C_{Cconfl})*(1 - n*C_2) \qquad (36)$$

We consider (36) when $1 > n*C_1$, $1 = n*C_1$ and $1 < n*C_1$

When $1 > n*C_1$ then from (36) and (31) we obtain:

$$C_2 < C_1 \qquad (37)$$

Thus when $S_C > n*S_L$ then $Q_c > Q_d$ only when $C_2 < C_1$, i.e., when D-

DBS and C-DBS use different CC mechanism or they use the same one but the D-DBS local processor throughput decrease due to synchronization of nonconflicting transactions is smaller than the decrease in local processor transaction processing rate compared to C-DBS transaction processing rate.

When $1 = n*C_1$, i.e., when $S_C = n*S_L$ then (36) always holds and thus $Q_c > Q_d$.

When $1 < n*C_1$ then from (36) and (31) we get

$$C_2 > C_1 \tag{38}$$

Thus when $S_C < n*S_L$ then $Q_c > Q_d$ only if $C_2 > C_1$. If $C_2 < C_1$ then $Q_d > Q_c$. The implication here is that $Q_d > Q_c$ if either D-DBS uses more efficient CC mechanism than C-DBS or D-DBS uses the same CC mechanism but its implementation is more efficient compared to D-DBS transaction processing, i.e., if

$$\frac{S_{Lcc}}{S_{Ccc}} < \frac{S_L}{S_C}$$

As can be seen from (36), (37) and (38) CC mechanism is a significant issue when comparing performance of C-DBS and D-DBS systems. Moreover it is important even for the performance of D-DBS based on fast network and for applications where either there are few interferring transactions or where there are few global transactions.

5. Conclusions.

In this paper we have investigated transaction execution cost and system throughput in C-DBS compared to D-DBS systems based on slow and fast networks. The conclusions reached in this paper indicate that the efficiency of CC mechanism is of great importance when comparing C-DBS and fast or slow D-DBS throughput. The same observation applies when comparing transaction execution cost in C-DBS and slow D-DBS. It seems that CC mechanism is not important when comparing the cost of transaction execution in C-DBS and fast D-DBS.

In this paper we have also indicated for which applications any D-DBS is likely to be a better solution than any C-DBS and vice versa.

6. Acknowledgement.

The author would like to acknowledge support of the NPS Foundation Research Programs for this work.

References

[BAD80]

Badal, D. Z., "The Analysis of the Effects of Concurrency Control on Distributed Databse System Performance", Proceedings of the 6th International Conference on Very Large Data Bases, Montreal, October 1980, 376-384.

[BAD81]

Badal, D. Z., "Concurrency Control Overhead or Closer Look at

Blocking vs. Nonblocking Concurrency Control Mechanisms", Proceedings of the 5th Berkeley Conference on Distributed Data Management and Computer Networks, San Francisco, February 1981.

[BUC79]

Bucci , G. and Streeter, D. N., " A Methodology for the Design of Distributed Information Systems", CACM 22, 4 (April 1979), 233-245.

[MOL79]

Garcia-Molina, H., "Performance of Update Algorithms for Replicated Data in a Distributed Database," Ph.D. dissertation, Department of Computer Science, Stanford University, June 1979.

[RIE79]

Ries, D. R., "The Effects of Concurrency Control on Database Management System Performance," Ph.D. dissertation, Computer Science Department, University of California, Berkeley, April 1979.

[STE79]

Stewart, H. M. "Performance analysis of Complex Communications Systems", IBM System Journal 18, 3 (1979), 356-373.

INITIAL DISTRIBUTION LIST

Defense Technical Information Center Cameron Station Alexandria, VA 22314	2
Dudley Knox Library Code 0142 Naval Postgraduate School Monterey, CA 93940	2
Office of Research Administration Code 012A Naval Postgraduate School Monterey, CA 93940	1
Chairman, Code 52Hq Department of Computer Science Naval Postgraduate School Monterey, CA 93940	2
Dushan Badal, Code 52Zd Department of Computer Science Naval Postgraduate School Monterey, CA 93940	5
Robert B. Grafton Office of Naval Research Code 437 800 N. Quincy Street Arlington, VA 22217	1
David W. Mizell Office of Naval Research 1030 East Green Street Pasadena, CA 91106	1
Vinton G. Cerf DARPA/IPTO 1400 Wilson Blvd Arlington, VA 22209	1
CDR R. Ohlander DARPA 1400 Wilson Blvd Arlington, VA 22209	1
Col. D. Adams DARPA 1400 Wilson Blvd Arlington, VA 22209	1

CAPT. W. Price 1
AFOSR/NM
Bolling AFB., D.C. 20332

Col. R. Schell 1
National Security Agency
C1
Fort George Meade, MD 20755

Raymond A. Liuzzi 1
RADC/COTD
Griffiss AFB., N.Y. 13441

Thomas Lawrence 1
RADC/COTD
Griffiss AFB., N.Y. 13441